**Denoising Autoencoder**

Zheng Lu (ECE692)

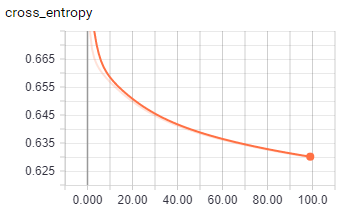
## Task 1:

1. Common settings for experiments of task 1:

* hidden layer size: 1024
* batch size 64
* epochs 100
* learning rate 0.05

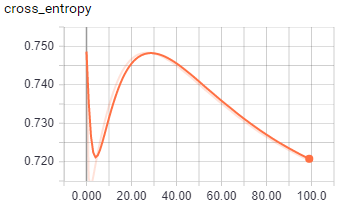
1. Out-of-box (yaldt) experiment running on cifar10:

* corruption module none
* activation function sigmoid
* loss function cross entropy
* stochastic gradient descent without momentum



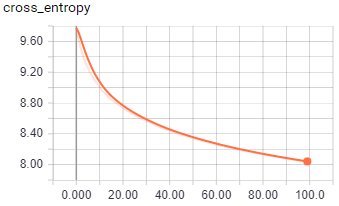
1. Add Gaussian noise as corruption module:

* Same as 1 except:
* Corruption module: Gaussian



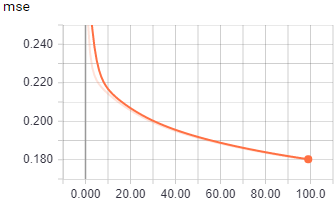
1. Replace sigmoid with ReLu:

* Same as 1 except:
* activation function ReLu



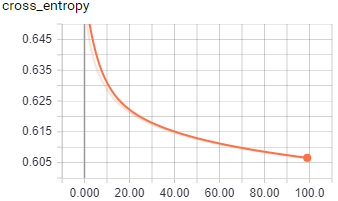
1. Replace cross entropy with squared loss:

* Same as 1 except:
* loss function squared loss



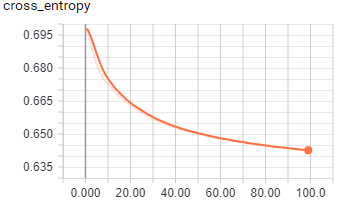
1. Add momentum:

* Same as 1 except:
* stochastic gradient descent with momentum (0.9)



1. Recommended settings:

* Same as 1 except:
* Uniform corruption module, sigmoid, cross entropy and momentum (0.9)



## Task 2:

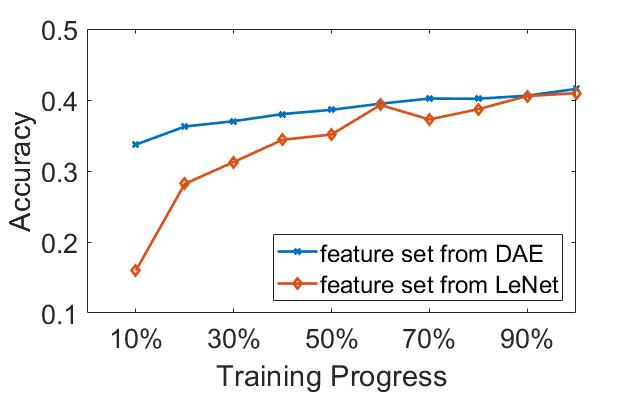
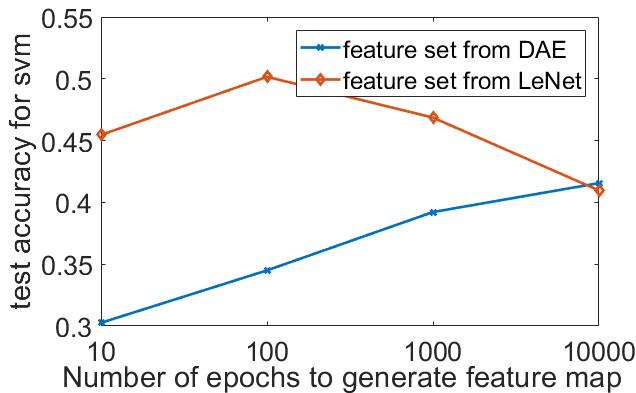
We use the training sample to train both the DAE and LeNet first for 1000 epoch. Then we run testing set and feed the features extracted from DAE and LeNet to a SVM classifier to get the testing accuracy and show below.

1. Settings for autoencoder:

* Same hidden layer size, batch size and learning rate as case 1 in task 1
* Run 10,000 epoch
* Gaussian corruption module
* ReLu activation function
* Cross entropy as loss function
* SGD with momentum

1. Setting for lenet

* Same hidden layer size, batch size and learning rate as case 1 in task 1
* Run 10,000 epoch
* Gaussian corruption module
* ReLu activation function
* Cross entropy as loss function
* SGD with momentum

1. Results:
   1. We show the testing accuracy for model at different training stage. Note that the x axis is the number of training samples used to train the model while the x axis is the testing accuracy of 5000 testing samples.  
        
      We can see in this figure that the feature set from DAE can lead to higher testing accuracy at very early stage of the training of SVM.
   2. We then change the number of epochs for training when generating the feature set with DAE and LeNet. Then we show the testing accuracy of SVM when using the generated feature sets.  
        
      We can see that although the feature set from LeNet outperforms feature set from DAE, as the number of epochs increasing, the performance of the feature set from LeNet does not increase anymore. We think this is caused by the overfitting problem. In this case, DAE is more robust against LeNet.

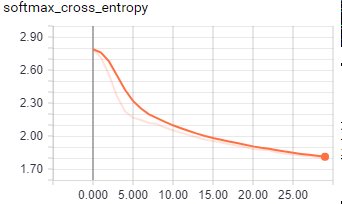
## Task 3

After many searches online, I found pretraining with unsupervised learning such as DAE is not a widely adopted method partly because of the implementation overhead and partly because that there is more powerful methods for CNNs such as batch normalization to speed up training process. So, instead of following the instruction of task 3 to use DAE to initialize LeNet, I add batch normalizations after each convolutional layer and fully connected layer to speed up the training process.

Settings:

* ReLu activation function
* Cross entropy as loss function
* SGD without momentum
* Learning rate of 1e-4
* Dropout at 0.5

Structure: conv2d-5-5-16-1, batch normalization, maxpool-2, conv2d-5-5-64-1, batch normalization, maxpool-2, full-1024, batch normalization, softmax

1. With batch normalization, the final test accuracy is 0.39 after 30 training epochs:  
   100 
   0200 
   11300 
   0400 
   0.00 
   0.000 5000 10.00 15 , 00 20 00 2500  
2. Without batch normalization, the final test accuracy is 0.198 after 30 training epochs:  
   accuracy 
   o. 190 
   0170 
   D 150 
   0130 
   0.000 
   20.00 2500 
   5000 
   1000 
   1500  softmax_cross_entropy 
   231 
   2.29 
   225 
   223 
   221 
   0 000 
   5000 
   10.00 
   1500 
   20.00 
   2500 

## Paper review (ECE692 only)

1. Supervised vs. unsupervised learning

Supervised method such as discriminative procedure like backpropagation ignores the structure in the input. It has bad performance when the output is more related to these structures than it is to the raw input. So, it makes sense to start by using unsupervised learning to discover latent variables that can model the structures in the ensemble of training images [1].

1. RBM and DBN  
   When the training set is large enough, the pretraining procedure seems not necessary. Recent researches on CNN with large dataset (ImageNet) have empirical evidence to support this claim. However, it is also argued that the unsupervised pretraining can provide better generalization of the network. I think as the trend right now is more and more data become public available, e.g. more and more researchers public dataset on recent top conferences in machine learning, the research on RBM and DBN are still going to be quiet in the foreseen future.
2. Variational Autoencoder  
   A generative model, can generate samples that does not exist in training set. Normally an autoencoder cannot generate representations, i.e., latent variables unless we feed an image to the input. This is done in variational autoencoder by forcing the latent variables learnt by the network follow a unit Gaussian distribution. After the network learnt the Gaussian distribution of the input, it can simply sample a latent vector from this distribution and generate a new image based on this latent vector [2].

The disadvantage is that the generated images are usually blurred.

## References

[1] Hinton, Geoffrey E. "To recognize shapes, first learn to generate images." *Progress in brain research* 165 (2007): 535-547.

[2] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114* (2013).